Zbornik radova Fakulteta tehničkih nauka, Novi Sad

UDK: 4.9
DOI: https://doi.org/10.24867/34BE30Jovin

KOROZ - *eBPF BAZIRANI OSVEŽIVAČ DNS CACHE-a*

KOROZ - eBPF BASED DNS CACHE REFRESHER

Vladimir Jovin, *Fakultet tehničkih nauka, Novi Sad*

Oblast – ELEKTROTEHNIKA I RAČUNARSTVO

Kratak sadržaj – *Ovaj rad istražuje upotrebu eBPF tehnologije za optimizaciju DNS saobraćaja. Implementacijom sistema baziranog na eBPF-u, XDP-u i Rust programskom jeziku, predstavljena je efikasna inspekcija i manipulacija mrežnim paketima u realnom vremenu. Prikazani su teorijski osnovi, opisane ključne komponente rešenja, rezultati i predloženi pravci daljeg razvoja, uključujući integraciju heuristika i veštačke inteligencije za optimizaciju DNS keš mehanizma.*

Ključne reči: *dns, ebpf, rust, ddi*

Abstract – *This paper explores the use of eBPF technology for DNS traffic optimization. By implementing a system based on eBPF, XDP, and the Rust programming language, real-time inspection and manipulation of network packets is demonstrated. Theoretical foundations are presented, key solution components are described, results are evaluated, and future development directions are proposed, including the integration of heuristics and artificial intelligence for DNS cache optimization.*

Keywords: *dns, ebpf, rust, ddi*

1. UVOD

U radu je predstavljena primena eBPF (Extended Berkeley Packet Filter) tehnologije za unapređenje performansi i sigurnosti DNS (Domain Name System) sistema. Korišćenjem XDP (Express Data Path) mehanizma i Aya razvojonog okvira u Rust-u, omogućena je brza i sigurna obrada DNS upita na nivou jezgra operativnog sistema.

Rad daje pregled teorijskih osnova, motivaciju za izbor tehnologija i opisuje mane tradicionalnih DNS rešenja. Cilj je detaljno prikazati implementaciju, analizu rezultata i smernice za dalji razvoj u oblasti optimizacije mrežnih protokola.

Korišćenjem XDP mehanizma, omogućava se efikasna inspekcija i obrada mrežnog saobraćaja pre nego što stigne do mrežnog interfejsa. Ova metoda ne samo da dozvoljava manipulaciju DNS upita i odgovora, već i omogućava dodatne mogućnosti za analizu i filtriranje saobraćaja.

NAPOMENA:
Ovaj rad proistekao je iz master rada čiji mentor je bio dr Veljko Petrović, docent

DNS serveri se suočavaju sa problemima u pogledu performansi i sigurnosti, što može dovesti do usporavanja mrežnih aplikacija i servisa. Korišćenje eBPF tehnologije omogućava implementaciju rešenja za optimizaciju DNS saobraćaja, uključujući inteligentno keširanje, prediktivno učitavanje i analizu obrazaca korišćenja u realnom vremenu.

Tehnologije za obradu mrežnog saobraćaja su ključne za unapređenje performansi i sigurnosti mrežnih sistema. Jedna od najznačajnijih inovacija u ovoj oblasti je eBPF (Extended Berkeley Packet Filter), tehnologija koja omogućava izvršavanje korisničkog koda unutar jezgra operativnog sistema, pružajući mogućnost za inspekciju i manipulaciju mrežnim paketima.

Rad je organizovan u pet ključnih poglavlja: uvod koji definiše problematiku DNS optimizacije, teorijski okvir koji temeljno obrazlaže DNS i eBPF koncepte, detaljna implementacija sistema "Koroz" sa tehničkim aspektima XDP obrade, evaluacija performansi sa konkretnim metrikama, te zaključak sa smernicama za dalja istraživanja.

2. DNS

DNS je sistem koji omogućava prevod ljudski čitljivih naziva domena u IP adrese, čime se olakšava pristup resursima na internetu [1; 2]. Sistem je razvijen u ranim 1980-im godinama kao odgovor na rastući broj računara povezanih na internet [3]. Pre DNS-a, korišćen je centralizovani *hosts.dat* fajl koji je sadržavao informacije o svim računarima na mreži, što je postalo neodrživo sa rastom interneta.

DNS je prvi put definisan u RFC 1034 [1] i RFC 1035 [2], objavljenim u novembru 1987. godine. Ovi dokumenti su postavili temelje za hijerarhijsku strukturu DNS-a i osnovne funkcionalnosti koje se koriste i danas. DNS funkcioniše na osnovu hijerarhijske strukture koja se sastoji od root servera na vrhu, TLD (Top-Level Domain) servera, i autoritativnih servera za specifične domene.

DNS obezbeđuje nekoliko ključnih funkcionalnosti: prevođenje imena u IP adrese, distribuciju podataka kroz decentralizovanu strukturu, keširanje odgovora za poboljšanje performansi, i podršku za različite tipove zapisa koji omogućavaju kompleksne mrežne konfiguracije.

3. eBPF

eBPF predstavlja tehnologiju koja je evoluirala iz originalnog BPF protokola razvijenog u ranim 1990-im godinama [4]. Dok je originalni BPF omogućio filtriranje mrežnih paketa, eBPF je proširio ove mogućnosti daleko van mrežnog konteksta, omogućavajući izvršavanje sigurnog koda u različitim delovima jezgra operativnog sistema.

Arhitektura eBPF-a se sastoji od nekoliko ključnih komponenti. eBPF programi se pišu u specijalizovanom jeziku sličnom C-u i kompajliraju se u eBPF bajt kod. Ovi programi prolaze kroz rigorozan proces verifikacije koji osigurava da ne mogu dovesti do nevalidnih stanja ili neovlašćenog pristupa memoriji [5]. Verifikator analizira sve moguće putanje izvršavanja i garantuje da program neće pristupiti nedozvoljenim memorijskim lokacijama.

3.1. Povezanost između DNS-a i eBPF-a

Kombinacija DNS-a i eBPF-a daje mogućnosti za optimizaciju mreža. Korišćenjem eBPF-a, moguće je implementirati efikasne mehanizme za inspekciju DNS upita i odgovora u realnom vremenu, bez značajnog uticaja na performanse sistema. Ova tehnologija omogućava analizu DNS saobraćaja, identifikaciju obrazaca korišćenja, i implementaciju inteligentnih strategija keširanja.

eBPF programi mogu prikupljati detaljne statistike o DNS saobraćaju, uključujući informacije o najčešće traženim domenima, latencijama upita, i obrascima pristupa. Ovi podaci mogu biti korišćeni za optimizaciju DNS servera i implementaciju prediktivnog keširanja koji može poboljšati performanse mrežnih aplikacija.

Važno je napomenuti da je ovaj pristup potpuno agnostičan prema konkretnoj DNS implementaciji, što znači da može raditi sa različitim DNS serverima kao što su BIND, Unbound, ili PowerDNS, bez potrebe za modifikacijama postojećih sistema.

4. MOTIVACIJA I TEHNOLOGIJE

Razvoj mrežnih aplikacija zahteva optimizaciju performansi i sigurnosti. U velikim mrežama, opterećenje DNS servisa prati organske obrasce pri čemu su TTL vrednosti uglavnom tipične za različite tipove zapisa. Analiza krive opterećenja DNS servera pokazuje da se određeni upiti ponavljaju u kratkim vremenskim intervalima, često pre nego što isteknu njihovi TTL-ovi.

Ova observacija otkriva priliku za optimizaciju. Umesto da se čeka da TTL istekne prirodno, sistem može proaktivno da osveži zapise koji se često koriste, pre nego što isteknu. Ovaj pristup može drastično smanjiti latenciju za krajnje korisnike i poboljšati ukupne performanse sistema.

eBPF tehnologija pruža mehanizam za implementaciju ovakve optimizacije jer omogućava posmatranje DNS saobraćaja bez uticaja na postojeće sisteme, kao i donošenje odluka o tome koji zapisi treba da se osveže u realnom vremenu.

4.1. Aya radni okvir

Za razliku od tradicionalnih pristupa koji se oslanjaju na libbpf ili bcc, Aya je izgrađena od nule isključivo u Rust-u, koristeći samo libc biblioteku kao zavisnost za sistemske pozive [6].

Jedan od benefita Aya razvojnog okvira je korišćenje musl libc umesto GNU libc (glibc). GNU C Library je podrazumevana standardna biblioteka na većini Linux distribucija i poznata je po svojoj bogatoj funkcionalnosti i širokoj kompatibilnosti. Međutim, zbog svoje kompleksnosti, glibc zahteva dinamičko linkovanje što može stvoriti probleme sa kompatibilnošću između različitih verzija sistema. Dodatno, musl je implementacija standardne C biblioteke zasnovana na API-ju sistemskih poziva Linuxa, uključujući interfejse definisane u osnovnom jezičkom standardu, kao i široko prihvaćenim proširenjima, koja omogućava potpuno statičko linkovanje.

Aya takođe pruža podršku za BPF Type Format (BTF) što omogućava da eBPF programi kompajlirani za jednu verziju kernela rade na različitim verzijama kernela bez potrebe za rekompajliranjem. Ova funkcionalnost je kritična za distribuciju eBPF aplikacija u heterogenim okruženjima.

4.2. XDP

XDP (eXpress Data Path) je izabran kao primarna tehnologija za obradu paketa zbog svojih prednosti u odnosu na alternative kao što su TC (Traffic Control) i tradicionalno procesiranje kroz SKB (Socket Buffer) strukturu.

XDP omogućava izvršavanje eBPF programa na najranijem nivou mrežnog steka, direktno na nivou mrežnog interfejsa. Ovo omogućava inspekciju i obradu paketa pre nego što stignu do bilo kog dela mrežnog steka operativnog sistema, što rezultuje niskim latencijama i visokim performansama.

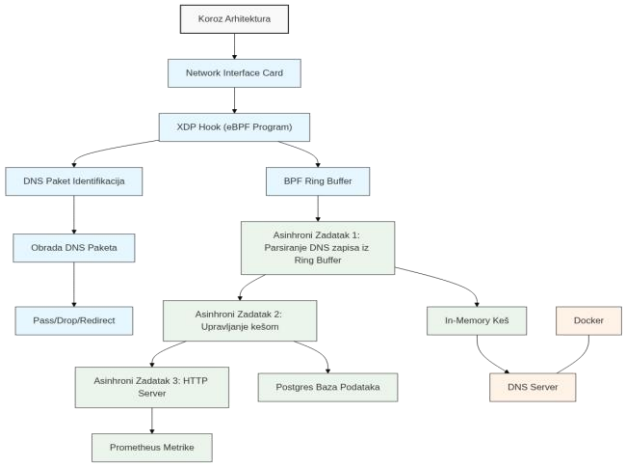
U poređenju sa TC (Traffic Control), koji omogućava upravljanje mrežnim saobraćajem na višem nivou u mrežnom steku, XDP pruža direktan pristup paketima bez potrebe za alokacijom SKB struktura. Ovo uklanja kompleksnost koja je povezana sa procesiranjem paketa.

SKB (Socket Buffer) je struktura podataka koja se koristi za skladištenje informacija o mrežnim paketima unutar jezgra. Iako je SKB veoma fleksibilan i podržava kompleksne operacije, njegova upotreba uvodi kompleksnost u obliku alokacije memorije i kopiranja podataka. XDP, s druge strane, radi direktno sa paketima u njihovoj originalnoj memorijskoj lokaciji, što uklanja potrebu za dodatnim kopiranjem podataka.

5. IMPLEMENTACIJA

Implementacija se sastoji od dve glavne komponente koje rade u koordinaciji: eBPF program koji se izvršava u jezgru operativnog sistema na XDP sklopci, i korisnički program koji upravlja prikupljenim podacima i implementira logiku za optimizaciju DNS keša (Slika 1).

eBPF program je odgovoran za inspekciju DNS saobraćaja koji prolazi kroz specifikovani mrežni interfejs. Program identifikuje DNS pakete analizirajući Ethernet i IP zaglavlja, a zatim UDP zaglavlje kako bi potvrdio da se radi o DNS saobraćaju (port 53). Kada se identifikuje relevantni paket, program ekstraktuje potrebne informacije i šalje ih u korisnički prostor putem BPF prstenastog bufera.



Slika 1: *Arhitektura sistema*

Korisnički program implementira složeniju logiku sistema kroz četiri glavna asinhrona zadatka.

Prvi zadatak kontinuirano čita podatke iz BPF prstenastog bufera, parsira DNS odgovore i prosleđuje ih za dalju obradu.

Drugi zadatak skladišti DNS odgovore u memorijski keš implementiran kao binarni hip, kao i u perzistentnu PostgreSQL bazu podataka za dugoročno čuvanje i analizu.

Treći zadatak implementira analizu DNS zapisa na osnovu njihovih TTL vrednosti i donošenje odluka o tome koji zapisi treba da se prethodno osveže. Ovaj zadatak periodično prolazi kroz keš, identifikuje zapise koji se približavaju isteku, i pokreće proces invalidacije i repopulacije kroz eksterni DNS resolver.

Četvrti zadatak implementira HTTP server koji omogućava pristup metrikama sistema za potrebe monitoringa (Prometheus), kao i REST API za pristup trenutnom stanju keša i statistikama.

5.1. BPF prstenasti bufer kao mehanizam komunikacije

BPF prstenasti bufer je izabran kao primarni mehanizam za komunikaciju između eBPF programa i korisničkog prostora zbog svojih značajnih prednosti u odnosu na alternative kao što je BPF *perf buffer* [7]. Prstenasti bufer predstavlja višestruki proizvođač, jednostruki potrošač (*MPSC*) red koji može bezbedno da se deli između više CPU-ova istovremeno.

U poređenju sa perf buferom, BPF prstenasti bufer nudi nekoliko ključnih prednosti. Prvo, efikasnije korišćenje memorije jer koristi jedan zajednički bafer umesto više bafera po CPU-u. Drugo, garantovane su bolje garancije redosleda događaja jer se svi događaji emituju u zajedničkom buferu umesto u *per-CPU* buferima koji se mogu potrošiti različitim brzinama.

Treće, prstenasti bufer podržava "*reserve/submit*" API koji omogućava rezervaciju prostora za podatke unapred, čime se izbegava dodatno kopiranje memorije i omogućava efikasnije korišćenje resursa. Ova funkcionalnost je posebno važna u visoko performantnim scenarijima gde svako kopiranje podataka može imati značajan uticaj na performanse.

5.2. Strategije optimizacije i heuristike

Sistem implementira strategiju za odlučivanje o tome koji DNS zapisi treba da se osveže. Osnovna heuristika se zasniva na analizi TTL vrednosti i obrasce pristupa zapisima. Zapisi koji se često koriste i čiji TTL se približava isteku imaju viši prioritet za osvežavanje.

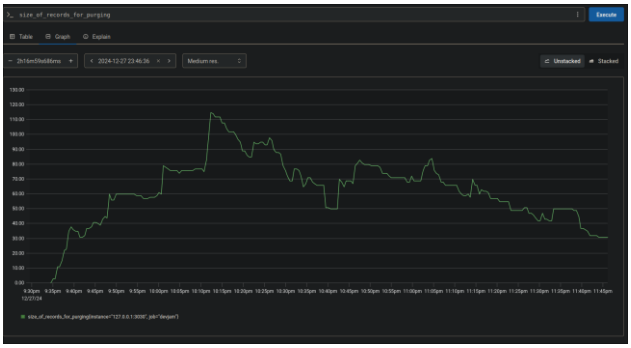
Algoritam koristi binarni hip strukturu podataka za efikasno održavanje zapisa sortiranih po vremenu isteka. Ova struktura omogućava *O(log n)* složenost za dodavanje novih zapisa i *O(1)* složenost za pristup zapisu koji će prvi da istekne. Periodično, zadatak prolazi kroz *heap* i identifikuje zapise koji se nalaze unutar konfigurabilnog vremenskog okvira pre isteka. Za zapise koji se identifikuju za osvežavanje, prvo se poziva proces invalidacije koji uklanja postojeći zapis iz DNS resolver keša. Nakon uspešne invalidacije, pokreće se proces repopulacije koji eksplicitno traži svežu verziju zapisa od autoritativnog servera.

Takođe su implementirane metrike za praćenje efikasnosti optimizacije, uključujući broj uspešnih invalidacija, broj neuspešnih operacija, latencije operacija, i ukupno vreme uštede za krajnje korisnike. Ove metrike omogućavaju kontinuirano podešavanje sistema i identifikaciju mogućnosti za dalje poboljšanje.

6. REZULTATI

Testiranje sistema je sprovedeno u kontrolisanom uobičajenom okruženju upotrebe tokom nekoliko dana kontinuiranog rada. Prometheus metrike pokazuju da sistem uspešno identifikuje i procesira DNS saobraćaj, sa dominantnim učešćem A (IPv4) i AAAA (IPv6) zapisa što je u skladu sa očekivanjima.

Analiza kardinaliteta reda za invalidaciju pokazuje stabilne vrednosti sa maksimumom od 115 zapisa u posmatranom periodu i prosekom značajno ispod tog nivoa (Slika 2).



Slika 2. *Kardinaliet reda zapisa u memoriji*

Metrike o broju invalidiranih i repopuliranih zapisa pokazuju obrasce DNS saobraćaja. Vidljivo je da dominiraju A (IPv4) rekordi, koji su za 2 sata dostigli 14000 akcija. Sistem demonstrira sposobnost da prati varijacije u opterećenju i prilagodi svoje operacije u skladu sa tim (Slika 3).

Slika 3: *Akcije nad zapisima*

Posebno značajna je observacija da implementacija održava nizak nivo neuspešnih operacija, što ukazuje na robusnosti efikasno rukovanje greškama. Ova karakteristika je kritična za produkcijska okruženja.

6.1. Pravci budućeg razvoja

Jedan od pravaca budućeg razvoja je proširenje sistema na korišćenje proizvoljnih heuristika i integracija AI modela. Ideja je da se sistem transformiše u otvorenu platformu koja klijentima omogućava implementaciju proizvoljne logike za optimizaciju putem REST API-ja.

Ovaj pristup bi omogućio kreiranje ekosistema gde različite komponente mogu da dele svoje heuristike i strategije optimizacije, što bi dovelo do poboljšanja efikasnosti DNS sistema. AI modeli bi mogli da analiziraju obrasce u DNS saobraćaju i predvide koji zapisi će verovatno biti potrebni u budućnosti, omogućavajući proaktivnost.

Drugi značajan pravac razvoja je dodatno pomeranje logike iz korisničkog prostora u eBPF program. Kako se eBPF tehnologija dalje razvija i podržava sve kompleksnije operacije, postaje moguće implementirati logiku direktno u jezgru. Ovo bi moglo da poboljša performanse eliminišući potrebu za komunikacijom između jezgra i korisničkog prostora za mnoge operacije.

7. ZAKLJUČAK

Ovaj rad demonstrira primenu eBPF tehnologije za optimizaciju DNS saobraćaja kroz implementaciju koja posmatra DNS upite i odgovore i proaktivno osvežava DNS keš na osnovu uočene potrebe. Kombinacija eBPF-a, XDP-a, Aya razvojnog okvira i Rust programskog jezika pokazala se kao adekvatna platforma za razvoj mrežnih aplikacija.

Predstavljene su praktične primene eBPF tehnologije za DNS optimizaciju, implementacija efikasnog sistema za analizu mrežnog saobraćaja u realnom vremenu, i razvoj arhitekture koja može da se prilagodi različitim okruženjima i zahtevima.

Integracija eBPF-a sa DNS-om omogućava razvoj naprednih rešenja koja poboljšavaju performanse, sigurnost i efikasnost mrežnih usluga. Ova sinergija predstavlja značajan korak napred u optimizaciji mrežnog saobraćaja i unapređenju korisničkog iskustva. Korišćenjem eBPF-a, XDP-a i Rust-a, moguće je implementirati efikasne mehanizme za inspekciju i manipulaciju mrežnim paketima u realnom vremenu. Dalji razvoj može dodatno unaprediti ove elemente i omogućiti još složenije i efikasnije operacije direktno u jezgru operativnog sistema.

8. LITERATURA

[1] RFC 1034 - DOMAIN NAMES - CONCEPTS AND FACILITIES
https://www.rfc-editor.org/rfc/rfc1034
(pristupljeno u decembru 2024.)

[2] RFC 1035 - DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION
https://www.rfc-editor.org/rfc/rfc1035
(pristupljeno u decembru 2024.)

[3] Hon K 2024 Web, URLs, domains, DNS Technology and security for lawyers and other professionals (Edward Elgar Publishing) pp 317–36

[4] eBPF
https://ebpf.io/what-is-ebpf/
(pristupljeno u decembru 2024.)

[5] eBPF Verifier
https://ebpf.io/what-is-ebpf/#verification
(pristupljeno u decembru 2024.)

[6] Aya framework
https://aya-rs.dev/
(pristupljeno u decembru 2024.)

[7] BPF Ring Buffer
https://docs.kernel.org/bpf/ringbuf.html
(pristupljeno u decembru 2024.)

Kratka biografija:

Vladimir Jovin rođen je u Novom Sadu 1999. godine. Osnovu školu i gimnaziju završio je u Novom Sadu. Diplomski rad na Fakultetu tehničkih nauka u Novom Sadu odbranio je 2022. godine.

kontakt: me@dovla.rs